APPENDIX A

```
'Oex2000 - Dbs01
'Revision 2.0.0
'Body Detection and Measurement Algorithms

'Written by David B. Scott

Option Explicit
Dim fsa_cb  As Integer
Dim fsa_is As Integer
Dim fsa_bf As Integer
Dim fsa_tf As Integer
Dim fsa_ss As Integer
Dim PSW As Double
Dim PHW As Double
Dim Barray(0 To 2000) As Long
Dim NewData As Variant
'Storage for easier reference later
Dim NumRows As Long
Dim NumColumns As Long
Dim Stopit As Integer
Dim Head As Integer
Dim Feet As Integer
Dim FSASum As Double
Dim FSAAverage As Double
Dim FSASensors As Long
Dim ShoulderWidth As Double
Dim FSAWeight As Double
Dim FSAHeight As Double
Dim FSAIspring As Double
Dim datacall As Long
Const MARRAY As Integer = 10
Private Type Coefs
    'coefficients
    BFcoefa(1 To MARRAY) As Double
    CBcoefa(1 To MARRAY) As Double
    TFcoefa(1 To MARRAY) As Double
    IScoefa(1 To MARRAY) As Double
    SSprofa(1 To MARRAY) As Double
End Type
Dim carray As Coefs
Dim cindex As Integer
'Array transfer from Pad Data
Public Function Put_FSAData(ByVal element As Long, ByVal index As Long) As Double
    Dim i As Integer
    Dim x As Double
    On Error Resume Next
    Put_FSAData = -1
    If index < 0 Then Exit Function
    If index > 2000 Then Exit Function
    NewData = True
    x = 0
    Barray(index) = element
    If element Then datacall = datacall + 1
    For i = 0 To index
        x = x + Barray(i)
    Next i
    Put_FSAData = x
    If index = 1023 Then
        Crunchit
    End If
End Function
'send cb value when requested
Public Property Get CBcoef() As Variant
Dim atemp As Double
Dim j As Integer
On Error Resume Next
If NewData Then
    Call Crunchit
```

```
Dim StartPoint As POINTAPI
Dim EndPoint As POINTAPI
Dim MaxRow As Long
Dim MaxCol As Long
Dim Corner1 As POINTAPI
Dim Corner2 As POINTAPI
'Storage for tracking the mouse
Dim px As Long
Dim py As Long
'Storage for the zoom values
Dim XStart As Double
Dim YStart As Double
Dim XEnd As Double
Dim YEnd As Double
Dim StartRow As Long
Dim EndRow As Long
Dim StartCol As Long
Dim EndCol As Long
Dim Val As Double
Dim Distance As Long

'Auxdata window stuff
Dim AuxSensors As Long
Dim AuxSum As Double
Dim AuxWidth As Double
Dim AuxLength As Double
Dim AuxAverage As Double

'Overall window stuff
Dim FSASum As Double
Dim FSAAverage As Double
Dim FSASensors As Long
Dim TorsoAverage As Double
Dim TorsoSensors As Integer
Dim ShoulderAverage As Double
Dim ShoulderWidth As Double
Dim HipAverage As Double
Dim WaistAverage As Double
Dim HipMaxWidth As Double
Dim WAverageWidth As Double
Dim FSAWeight As Double
Dim FSAHeight As Double
Dim FSAIspring As Double
Dim TorsoLength As Double


Private Sub Chart3D1_DblClick()
'Capture the double click.

    DoubleClick = True
End Sub

Private Sub Chart3D1_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)
'Watch for the user to press the mouse button so we can create the
' data rectangle to use for the zoom process.

    'Make sure it is the left button and then get the needed information
    If Button = 1 And Shift = 0 Then

        Chart3D1.Refresh
        'Get the API information from the main Chart
```

```
        ChartDc = GetDC(C[redacted]3D1.hWnd)
        PenHandle = Creat[redacted](0, 2, QBColor(0))
        OldPenHandle = SelectObject(ChartDc, PenHandle)
        Result = SetROP2(ChartDc, vbNotXorPen)

        'Get the number of rows and columns in use
        MaxRow = Chart3D1.ChartGroups(1).ElevationData.RowCount
        MaxCol = Chart3D1.ChartGroups(1).ElevationData.ColumnCount

        'Get the pixel co-ordinates of the lower-left and upper-right corners of the
        ' main chart so we can constrain the "Rubber Band" to stay on the data area
        Chart3D1.ChartGroups(1).DataIndexToCoord 1, 1, Corner1.x, Corner1.y
        Chart3D1.ChartGroups(1).DataIndexToCoord MaxRow, MaxCol, Corner2.x, Corner2.y

        px = x / Screen.TwipsPerPixelX                  'Convert the mouse location to pixels
        py = y / Screen.TwipsPerPixelY

        'If we are outside the chart, set the values to be outside the allowable range
        If px < Corner1.x Or px > Corner2.x Then
            StartPoint.x = -1
            StartPoint.y = -1
            EndPoint.x = -1
            EndPoint.y = -1

            'Release the resources as we no longer need them
            Result = SelectObject(ChartDc, OldPenHandle)
            Result = DeleteObject(PenHandle)
            Result = ReleaseDC(Chart3D1.hWnd, ChartDc)
            Exit Sub
        End If
        If py < Corner2.y Or py > Corner1.y Then
            StartPoint.x = -1
            StartPoint.y = -1
            EndPoint.x = -1
            EndPoint.y = -1

            'Release the resources as we no longer need them
            Result = SelectObject(ChartDc, OldPenHandle)
            Result = DeleteObject(PenHandle)
            Result = ReleaseDC(Chart3D1.hWnd, ChartDc)
            Exit Sub
        End If

        'Set the startpoint of the rectangle to the current mouse position
        StartPoint.x = px
        StartPoint.y = py
        EndPoint.x = px
        EndPoint.y = py

        'Draw the "Rubber Band" rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        'Release the resources as we no longer need them
        Result = SelectObject(ChartDc, OldPenHandle)
        Result = DeleteObject(PenHandle)
        Result = ReleaseDC(Chart3D1.hWnd, ChartDc)

        Region = Chart3D1.ChartGroups(1).CoordToDataCoord(StartPoint.x, StartPoint.y, XStart, YS
tart, Val)
        Region = Chart3D1.ChartGroups(1).CoordToDataIndex(px, py, Row, Col, Distance) 'Get the D
ata Values at the current location
```

```
    NewData = False
End If
atemp = 0
For j = 1 To MARRAY
    atemp = atemp + carray.CBcoefa(j)
Next j
CBcoef = Format$(atemp / MARRAY, "0")
End Property
'send is val when requested
Public Property Get IScoef() As Variant
Dim atemp As Double
Dim j As Integer
On Error Resume Next
If NewData Then
    Call Crunchit
    NewData = False
End If
atemp = 0
For j = 1 To MARRAY
    atemp = atemp + carray.IScoefa(j)
Next j
IScoef = Format$(atemp / MARRAY, "0")
End Property
'send bf value when requested
Public Property Get BFcoef() As Variant
Dim atemp As Double
Dim j As Integer
On Error Resume Next
If NewData Then
    Call Crunchit
    NewData = False
End If
atemp = 0
For j = 1 To MARRAY
    atemp = atemp + carray.BFcoefa(j)
Next j
BFcoef = Format$(atemp / MARRAY, "0")
End Property
'send tf value when requested
Public Property Get TFcoef() As Variant
Dim atemp As Double
Dim j As Integer
On Error Resume Next
If NewData Then
    Call Crunchit
    NewData = False
End If
atemp = 0
For j = 1 To MARRAY
    atemp = atemp + carray.TFcoefa(j)
Next j
TFcoef = Format$(atemp / MARRAY, "0")
End Property
'send spine position when requested
Public Property Get SSprof() As Variant
Dim atemp As Double
Dim j As Integer
On Error Resume Next
If NewData Then
    Call Crunchit
    NewData = False
End If
atemp = 0
For j = 1 To MARRAY
    atemp = atemp + carray.SSprofa(j)
Next j
SSprof = Format$(atemp / MARRAY, "0")
End Property
```

```
'Actual Mathmatics for Body [redacted]ection and Measurements
Private Sub Crunchit()
    Dim r As Integer
    Dim c As Integer
    Dim InARow As Integer
    Dim DTemp As Double
    Dim fsa As Integer
    Dim darray(32, 32) As Double
    Dim Lumbar As Double
    Dim CrotchHeight As Integer
    Dim CData(1 To 32, 1 To 32)
    Dim UnitMultiplier As Double
    Dim Filter As Double
    Dim Center As Integer
    Dim Cfirst As Integer
    Dim TorsoCenter As Integer
    Dim WaistCenterSum As Integer
    Dim WaistAverage As Double
    Dim Filter2
    On Error Resume Next

    UnitMultiplier = 0.392156862745098

    fsa = 0
    FSASum = 0
    FSASensors = 0

    NumColumns = 32
    NumRows = 32

    Filter = 0.75
    Filter2 = 2
    'Decode one-dimensional data into three-dimensional form
    For c = 1 To NumColumns
        For r = 1 To NumRows

            DTemp = Barray(fsa) * UnitMultiplier
            CData(c, r) = DTemp
                 If DTemp > Filter Then
                    FSASum = FSASum + DTemp
                    If DTemp > Filter2 Then FSASensors = FSASensors + 1
                End If
             fsa = fsa + 1
          Next r
    Next c
    If FSASensors = 0 Then GoTo no_body
    'Compute Sensor Average Pressure
    FSAAverage = FSASum / FSASensors
    'Compute theoretical weight
    FSAWeight = FSASum * 0.0155
    ' if the person is less than 40 lbs - abort
    If FSAWeight < 40 Then GoTo no_body
    'set up first cb factor..
    fsa_cb = FSAWeight * 3.5
    'Find the Head or top active sensor
    Stopit = False
    For c = 1 To NumColumns
        For r = 1 To NumRows
            If CData(c, r) > Filter2 Then Stopit = True
            If Stopit Then Exit For
        Next r
        If Stopit Then Exit For
    Next c
    Head = c
    'Find the feet or bottom active sensor
    Stopit = False
    For c = NumColumns To 1 Step -1
        For r = NumRows To 1 Step -1
```

```
            If CData(c, r) > [redacted]lter2 Then Stopit = True
            If Stopit Then E[redacted] For
        Next r
        If Stopit Then Exit For
    Next c
    Feet = c
    'Calculate estimated Height based on Head & Feet detection
    FSAHeight = (2 + (Feet - Head)) * 2.25
    If FSAAverage = 0 Then GoTo error_out
'select crotch height/shoulder width based on calculated height
'from statistical ave values
Select Case (2 + (Feet - Head))      'note #of sensor rows not inches
    Case 25
        CrotchHeight = 11
        ShoulderWidth = 19
    Case 26
        CrotchHeight = 12
        ShoulderWidth = 20
    Case 27
        CrotchHeight = 12
        ShoulderWidth = 21
    Case 28
        CrotchHeight = 13
        ShoulderWidth = 22
    Case 29
        CrotchHeight = 13
        ShoulderWidth = 24
    Case 30
        CrotchHeight = 14
        ShoulderWidth = 24
    Case 31
        CrotchHeight = 14
        ShoulderWidth = 25
    Case 32
        CrotchHeight = 15
        ShoulderWidth = 26
    Case 33
        CrotchHeight = 15
        ShoulderWidth = 27
    Case 34
        CrotchHeight = 15
        ShoulderWidth = 27
    Case 34
        CrotchHeight = 15
        ShoulderWidth = 27
    Case Else
        CrotchHeight = 31
        ShoulderWidth = 31
End Select
    'inter spring value set based on fsa ave weight
    FSAIspring = FSAWeight / FSAAverage
    fsa_is = FSAIspring * 100
    'look for top of shoulders (not used at this time)
    Stopit = 0
    InARow = 0
    For c = 1 To 16
        If c > 32 Then GoTo error_out
        If c < 1 Then GoTo error_out
        For r = 1 To NumRows
            If CData(c, r) > Filter Then
                InARow = InARow + 1
                If InARow < Stopit Then Stopit = InARow
            Else: InARow = 0
            End If
        Next r
        If Stopit < 12 Then Exit For        '12 in a row is down past the head
        Stopit = 0
    Next c
```

```
    'look for center of shoulders (not used at this time)
    If c < 1 Then c = 1
    Center = 0
    Cfirst = 0
    Stopit = False
    For r = 1 To NumRows
        If CData(c, r) > Filter Then
            Center = Center + 1
            If Cfirst = 0 Then Cfirst = r
        End If
    Next r
    fsa_bf = (((FSAWeight / ShoulderWidth)) * 45)
    Center = 0
    Cfirst = 0
    Stopit = False
    'find center of hips
    c = CrotchHeight + 1
    For r = 1 To NumRows
        If CData(c, r) > Filter Then
            Center = Center + 1
            If Cfirst = 0 Then Cfirst = r
        End If
    Next r
    TorsoCenter = Cfirst + (Center / 2)
    WaistCenterSum = 0
    'look at the lumbar area for weight
    For c = CrotchHeight - 2 To CrotchHeight
        For r = TorsoCenter - 5 To TorsoCenter + 5
            If c - 4 < 1 Then GoTo error_out
                WaistCenterSum = WaistCenterSum + CData(c - 4, r)
        Next r
    Next c
    'find lumbar
    WaistAverage = (WaistCenterSum / 33) / FSAAverage
    Lumbar = WaistAverage
    fsa_tf = Lumbar * 100
    fsa_ss = 0
    If Lumbar > 1 Then fsa_ss = 1
    If Lumbar > 1.1 Then fsa_ss = 2
    If Lumbar > 1.2 Then fsa_ss = 3
    If Lumbar > 1.3 Then fsa_ss = 4
    If Lumbar > 1.4 Then fsa_ss = 5
    If Lumbar > 1.5 Then fsa_ss = 6
    If Lumbar > 1.6 Then fsa_ss = 7
    If Lumbar > 1.7 Then fsa_ss = 8
    If Lumbar > 1.8 Then fsa_ss = 9
    GoTo end_sub
'if min wieght is not meet the return 0's
no_body:
    fsa_cb = 0
    fsa_is = 0
    fsa_bf = 0
    fsa_tf = 0
    fsa_ss = 0

    GoTo end_sub
'if error return default numbers
error_out:
    fsa_cb = 300
    fsa_is = 500
    fsa_bf = 255
    fsa_tf = 100
    fsa_ss = 5
end_sub:
    carray.CBcoefa(cindex) = fsa_cb
    carray.IScoefa(cindex) = fsa_is
    carray.BFcoefa(cindex) = fsa_bf
    carray.TFcoefa(cindex) = fsa_tf
```

```
    carray.SSprofa(cindex) = fsa_ss
    cindex = cindex + 1
    If cindex > MARRAY Then cindex = 1
    Exit Sub
 End Sub
Private Sub Class_Initialize()
    'setup coefficent arrays to 0
    Dim i As Integer
    For i = 1 To MARRAY
        carray.BFcoefa(i) = 0
        carray.CBcoefa(i) = 0
        carray.IScoefa(i) = 0
        carray.SSprofa(i) = 0
        carray.TFcoefa(i) = 0
    Next i
    cindex = 1
End Sub
```

```
        'Store the row and column values for use in the sub-set creation later
        StartRow = Row
        StartCol = Col

    End If
End Sub
Private Sub Chart3D1_MouseMove(Button As Integer, Shift As Integer, x As Single, y As Single)
'Track the movement of the mouse and update the "Rubber Band" rectangle.

    If Button = 1 And StartPoint.x <> -1 And Shift = 0 Then

        'Get the API information from the main Chart
        ChartDc = GetDC(Chart3D1.hWnd)
        PenHandle = CreatePen(0, 2, QBColor(0))
        OldPenHandle = SelectObject(ChartDc, PenHandle)
        Result = SetROP2(ChartDc, vbNotXorPen)

        'Get rid of the old rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        'Convert the screen co-ordinates to pixels
        px = x / Screen.TwipsPerPixelX
        py = y / Screen.TwipsPerPixelY

        'Constrain the "Rubber Band" rectangle to stay on the data area of the chart
        If px >= Corner1.x And px <= Corner2.x Then
            EndPoint.x = px
        Else
            If px < Corner1.x Then
                EndPoint.x = Corner1.x
            Else
                EndPoint.x = Corner2.x
            End If
        End If
        If py >= Corner2.y And py <= Corner1.y Then
            EndPoint.y = py
        Else
            If py > Corner1.y Then
                EndPoint.y = Corner1.y
            Else
                EndPoint.y = Corner2.y
            End If
        End If

        'Draw the new rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        'Release the resources as we no longer need them
        Result = SelectObject(ChartDc, OldPenHandle)
        Result = DeleteObject(PenHandle)
        Result = ReleaseDC(Chart3D1.hWnd, ChartDc)
    End If
End Sub

Private Sub Chart3D1_MouseUp(Button As Integer, Shift As Integer, x As Single, y As Single)
'Capture the mouse up event so we know when the user is done creating the rectangle.
'Copy the current graph to one of the empty locations, and then perform the zoom.

    Static i As Integer
    Static J As Integer
```

```
    Dim hd As Integer
    Dim hld As Boolean

    hd = -1

    'Check and make sure there is data to zoom in on first and exit if there isn't any selected
    If Abs(StartPoint.x - EndPoint.x) < 1 Or Abs(StartPoint.y - EndPoint.y) < 1 Then

        'Get the API information from the main Chart
        ChartDc = GetDC(Chart3D1.hWnd)
        PenHandle = CreatePen(0, 2, QBColor(0))
        OldPenHandle = SelectObject(ChartDc, PenHandle)
        Result = SetROP2(ChartDc, vbNotXorPen)

        'Clear the rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        'Release the resources as we no longer need them
        Result = SelectObject(ChartDc, OldPenHandle)
        Result = DeleteObject(PenHandle)
        Result = ReleaseDC(Chart3D1.hWnd, ChartDc)
        Exit Sub
    End If

    If Button = 1 And StartPoint.x <> -1 Then

        'Get the API information from the main Chart
        ChartDc = GetDC(Chart3D1.hWnd)
        PenHandle = CreatePen(0, 2, QBColor(0))
        OldPenHandle = SelectObject(ChartDc, PenHandle)
        Result = SetROP2(ChartDc, vbNotXorPen)

        'Get rid of the old rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        px = x / Screen.TwipsPerPixelX              'Convert screen co-ordinates to pixels
        py = y / Screen.TwipsPerPixelY

        'Constrain the "Rubber Band" rectangle to the data area of the chart
        If px >= Corner1.x And px <= Corner2.x Then
            EndPoint.x = px
        Else
            If px < Corner1.x Then
                EndPoint.x = Corner1.x
            Else
                EndPoint.x = Corner2.x
            End If
        End If
        If py >= Corner2.y And py <= Corner1.y Then
            EndPoint.y = py
        Else
            If py > Corner1.y Then
                EndPoint.y = Corner1.y
            Else
                EndPoint.y = Corner2.y
            End If
        End If

        'Draw the new rectangle
        Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)
```

```
        'Capture values fo[redacted]e in the creation of the subset[redacted]er
        Region = Chart3D1.[redacted]rtGroups(1).CoordToDataCoord(EndPoint.x, EndPoint.y, XEnd, YEnd, Va
l)
        Region = Chart3D1.ChartGroups(1).CoordToDataIndex(EndPoint.x, EndPoint.y, Row, Col, Dist
ance) 'Get the Data Values at the current location

        If Region = oc3dRegionInChartArea Then
            EndRow = Row
            EndCol = Col
        End If


        'Clear the rectangle
'       Result = Rectangle(ChartDc, StartPoint.x, StartPoint.y, EndPoint.x, EndPoint.y)

        'Release the resources as we no longer need them
        Result = SelectObject(ChartDc, OldPenHandle)
        Result = DeleteObject(PenHandle)
        Result = ReleaseDC(Chart3D1.hWnd, ChartDc)

        Debug.Print StartRow; StartCol; EndRow; EndCol

        'Switch around the rows and cols to make things easier
        If StartRow > EndRow Then
            i = StartRow
            StartRow = EndRow
            EndRow = i
        End If

        If StartCol > EndCol Then
            i = StartCol
            StartCol = EndCol
            EndCol = i
        End If

        AuxSensors = 0
        AuxSum = 0

        For i = StartCol To EndCol
            For J = StartRow To EndRow
                If Chart3D1.ChartGroups(1).ElevationData.Value(J, i) > 0 Then
                    AuxSum = AuxSum + Chart3D1.ChartGroups(1).ElevationData.Value(J, i)
                    AuxSensors = AuxSensors + 1
                End If
            Next J
        Next i

        If Len(CommonDialog1.filename) > 0 Then
            AuxWidth = Abs(YEnd - YStart)
            AuxLength = Abs(XEnd - XStart)
            AuxAverage = AuxSum / AuxSensors


            'Reset the location of the highlighted area

        End If
    End If
End Sub


Private Sub Command1_Click()
```

```
    If Me.BFcoef = 0 Or Me.SBcoef = 0 Or Me.IScoef = 0 Or Me.TFcoef = 0 Then
        EntryError.ErrorText.Caption = "You must have a FSA file or manually entered data to pro
ceed"
        EntryError.Show 1
        Exit Sub
    End If


    Main.Show 1
End Sub


Private Sub Command2_Click()
    GoExcel
End Sub

Private Sub Form_Load()
'This is where it all begins!

    Frame2.Enabled = False

    'Start with the form in the top-left corner
    Me.Top = 50
    Me.Left = 50

    'Setup the need variables
    Dim r As Integer
    Dim c As Integer
    Dim AxisValue As Double
    Dim delta As Double

    'These are backwards so because the grid is the main problem
    ' to contend with due to all the inversion necessary
    NumRows = Chart3D1.ChartGroups(1).ElevationData.ColumnCount
    NumColumns = Chart3D1.ChartGroups(1).ElevationData.RowCount

    'Set a default value
    DoubleClick = False

    'This puts the values of the Grid Index points into the header row
    ' of the grid control.  (grid control is at the bottom of the window)
    'NOTE: Becuase of the rotation of the graph, the columns and rows are
    '      reversed in order to fill the grid in correspondance with the graph
    delta = Chart3D1.ChartGroups(1).ElevationData.RowDelta(1)
    AxisValue = Chart3D1.ChartGroups(1).ElevationData.RowOrigin

    'This puts the values of the Grid Index points into the header column
    ' of the grid control. (grid control is at the bottom of the window)
    'NOTE: Becuase of the rotation of the graph, the columns and rows are
    '      reversed in order to fill the grid in correspondance with the
    '      chart
    delta = Chart3D1.ChartGroups(1).ElevationData.ColumnDelta(1)
    AxisValue = Chart3D1.ChartGroups(1).ElevationData.ColumnOrigin

    'Change the color of the Grid Lines
    Chart3D1.ChartArea.Axes("X").MajorGrid.Style.Color = ocColorCornflowerBlue
    Chart3D1.ChartArea.Axes("Y").MajorGrid.Style.Color = ocColorCornflowerBlue
    Chart3D1.ChartArea.Axes("Z").MajorGrid.Style.Color = ocColorCornflowerBlue
```

```
End Sub

Private Sub Form_Unload(Cancel As Integer)
'End the program.

    End
End Sub



Private Sub mnuAbout_Click()
'User wants to see what to do in this demo.

    With CommonDialog1
        .HelpCommand = cdlHelpContext
        .HelpContext = 18
        .HelpFile = App.HelpFile
        .ShowHelp
    End With
End Sub

Private Sub mnuAboutOlectra_Click()
'User wants to see what Olectra Chart 3D is all about.

    With CommonDialog1
        .HelpCommand = cdlHelpContext
        .HelpContext = 19
        .HelpFile = App.HelpFile
        .ShowHelp
    End With
End Sub

Private Sub mnuExit_Click()
'Exit the program.

    Unload Me
End Sub



Private Sub mnuOpen_Click()
    Dim sFile As String

    With CommonDialog1
        .filename = ""
        .Flags = 0
        'To Do
        'set the flags and attributes of the
        'common dialog control
        .Filter = "FSA Files (*.FSA)|*.*"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
        sFile = .filename
    End With

'    cancel = CommonDialog1.Action
    Crunchit (sFile)
```

```
End Sub
Private Sub Crunchit(sFile As String)

    Dim r As Integer
    Dim c As Integer
    Dim RAv As Double
    Dim avrav As Double
    Dim InARow As Integer
    Dim TorsoBottom As Integer
    Dim TorsoTop As Integer
    Dim TorsoLeft As Integer
    Dim TorsoRight As Integer
    Dim Brow As Integer
    Dim HipSum As Double
    Dim WaistSum As Double
    Dim HipSensors As Integer
    Dim WFirst As Integer
    Dim WLast As Integer
    Dim LastInARow As Integer
    Dim ShoulderSum As Double
    Dim ShoulderSensors As Integer
    Dim SLast As Integer
    Dim SFirst As Integer
    Dim Stretch As Integer
    Dim WaistSensors As Integer
    Dim DTemp As Double
    Dim TorsoCenter As Double
    Dim test As Variant

    ReadFile (sFile)

    Chart3D1.IsBatched = True

    fsa = 0
    FSASum = 0
    FSASensors = 0

    'fill in the chart region - batched

    For c = 1 To NumColumns
        For r = 1 To NumRows

            DTemp = VistaFile.FSAData(fsa) * VistaFile.UnitMultiplier

            Oex2000.Put_FSAData VistaFile.FSAData(fsa), fsa

            Chart3D1.ChartGroups(1).ElevationData.Value(c, r) = DTemp

            fsa = fsa + 1
        Next r

    Next c

    Chart3D1.IsBatched = False

    BFcoef.Text = Oex2000.BFcoef

    CBcoef.Text = Oex2000.CBcoef

    TFcoef.Text = Oex2000.TFcoef
```

```
    IScoef.Text = Oex2000[redacted]coef
    SSprof.Text = Oex2000.SSprof
    Dim holder1, holder2, holder3 As Double
    If Option1(0).Value = True Then
        HeadZone.Text = Format(Oex2000.FSAWeight * 0.11, "00")
        holder1 = (Oex2000.FSAWeight * 0.55)
        holder2 = Oex2000.ShoulderAverage + Oex2000.WaistAverage + Oex2000.HipAverage
        holder3 = 100 - holder2
        If holder3 < 0 Then holder3 = -holder3
        holder3 = holder3 / 2
        ShoulderZone.Text = Format(holder1 * ((Oex2000.ShoulderAverage + holder3) / 100), "00")
        WaistZone.Text = Format(holder1 * ((Oex2000.WaistAverage) / 100), "00")
        HipZone.Text = Format(holder1 * ((Oex2000.HipAverage + holder3) / 100), "00")
        ThighZone.Text = Format(Oex2000.FSAWeight * 0.2, "00")
        FeetZone.Text = Format(Oex2000.FSAWeight * 0.1, "00")

    Else


    End If
    frmSurface.Caption = "L&P Controls FSA Statistic Tool - " + sFile

End Sub

Private Sub Option1_Click(index As Integer)
    If index = 1 Then Frame2.Enabled = True Else: Frame2.Enabled = False


End Sub

Private Sub Utility_Click()
    Main.Show 1


End Sub
```

```
Public Sub GoExcel()
    Set xlApp = CreateObject("Excel.Application")

    Set xlBook = xlApp.Workbooks.Add

    Set xlSheet = xlBook.Worksheets(1)

    xlApp.Visible = True

    xlSheet.Cells(1, 1) = "Subject"
    xlSheet.Cells(1, 2) = "CB"
    xlSheet.Cells(1, 3) = "IS"
    xlSheet.Cells(1, 4) = "BF"
    xlSheet.Cells(1, 5) = "TF"

    Dim sFile As String
    Dim i As Integer


     For i = 1 To 79
        If i < 10 Then
        sFile = "C:\FSADATA\0" & i & ".fsa"
        Else: sFile = "C:\FSADATA\" & i & ".fsa"
        End If
        Crunchit (sFile)
        xlSheet.Cells(i + 1, 1) = i
        xlSheet.Cells(i + 1, 2) = Format$(Oex2000.CBcoef, "0")
        xlSheet.Cells(i + 1, 3) = Format$(Oex2000.IScoef, "0")
        xlSheet.Cells(i + 1, 4) = Format$(Oex2000.BFcoef, "0")
        xlSheet.Cells(i + 1, 5) = Format$(Oex2000.TFcoef, "0")

    Next i

    xlSheet.SaveAs ("FSAconversion")

    xlApp.Quit

End Sub
```

```
Private Done As Integer

Private Sub Form_Load()

    Done = False
    DoEvents
    Call ApiHome

    Done = True

End Sub

Private Sub Timer1_Timer()

    Dim Complete As Long

   If ProgressBar1.Value = 100 Then

        ProgressBar1.Value = 0
        Timer1.Enabled = False
        Complete = ApiComplete(0, "HOME")
        If Complete = &HAABFF Then
            Status.Caption = "Complete"
            Unload Me
            Exit Sub
        End If

        Timer1.Enabled = True

   End If

   ProgressBar1.Value = ProgressBar1.Value + 1
End Sub
```

APPENDIX B

```
Option Explicit
Dim PHW As Double
Dim PSW As Double

Dim xlApp As Excel.Application

Dim xlBook As Excel.Workbook

Dim xlSheet As Excel.Worksheet

Dim X1 As Double
Dim X2 As Double
Dim x3 As Double
Dim x4 As Double
Dim x5 As Double
Dim x6 As Double
Dim x7 As Double
Dim x8 As Double
Dim x9 As Double

Dim SpineData As Double
Dim Stopit As Integer
Dim Head As Integer
Dim Feet As Integer

'Indices of last grid index selected
Dim LastRow As Long
Dim LastCol As Long
Const NumHold As Integer = 3
'Indices of the current grid index being dragged
Dim PickRow As Long
Dim PickCol As Long

'Storage for easier reference later
Dim NumRows As Long
Dim NumColumns As Long

'True when rotating, etc.
Dim IsModifying As Boolean

'Keeps track of the region the mouse is in
Dim Region As Long
Dim OldRegion As Long

'Keeps track of the current row and column the mouse is on
Dim Row As Long
Dim Col As Long

'ASCII Character constants
Const CharEnter As Integer = 13

'Capture any double-clicks the user does
Dim DoubleClick As Boolean

Public fsa As Long
'Storage for drawing the zoom rectangle
Dim Result As Long
Dim PenHandle As Long
Dim OldPenHandle As Long
Dim ChartDc As Long
```

```
'

Option Explicit

Private Sub CancelButton_Click()
    Main.Choice = 1
    Unload Me

End Sub

Private Sub OKButton_Click()
    Main.Choice = 0
    Unload Me

End Sub
```

```
Public Project As Variant
Public ProjectDate As Variant
Public Choice As Integer
Public Head As Variant
Public Trunk As Variant
Public Thighs As Variant
Public Legs As Variant
Public Feet As Variant
Public TestStage As Variant
Public MonitorStage As Variant

Private Type Oexrecord
    ' identity stuff
    Description As String
    pDate As String
    Setup As Integer
    ' coefficients
    BFcoef As Double
    CBcoef As Double
    TFcoef As Double
    IScoef As Double
    SSprof As Double
    ' zone stuff
    Hdzone As Double
    Szone As Double
    Wzone As Double
    Hpzone As Double
    Tzone As Double
    Fzone As Double
    ' positional feedback
    Position(1 To 20) As Single
    ' calibration stuff
    Calibration(1 To 20) As Single
End Type

Private oexdata As Oexrecord


' Home Button
Private Sub Command1_Click()

    Home.Show 1

End Sub
' Calibrate Button
Private Sub Command3_Click()

    Calibrate.Show 1

End Sub
' Reset Button
Private Sub Command4_Click()

    Reset.Show 1

End Sub



' Test Button
Private Sub Command5_Click()
```

```
    Dim filename As Variant
    Dim i As Integer

    If ProjectText.Text = "" Then

        EntryError.ErrorText.Caption = "No Project Entered"
        EntryError.Show 1
        Exit Sub

    End If

    ' build the filename with directory and extension

    filename = "\oex2000\" + ProjectText.Text + ".oex"

    ' verify the existance or lack thereof

    If Dir(filename) = "" Then


    Else

        ' file already exists - replace = 0, cancel = 1
        FileError.Show 1

        If Main.Choice = 1 Then Exit Sub

    End If

    ' if we get here we have an open file "filename"
    ' and the data has been justified enough to proceed with the test

    If Option1(0).Value = True Then

        ApiHeadZone frmSurface.HeadZone.Text * 0.0151
        ApiShoulderZone (frmSurface.ShoulderZone.Text * 0.0151) / 3
        ApiWaistZone (frmSurface.WaistZone.Text * 0.0151) / 4
        ApiHipZone (frmSurface.HipZone.Text * 0.0151) / 2
        ApiThighZone (frmSurface.ThighZone.Text * 0.0151) / 2
        ApiFeetZone (frmSurface.FeetZone.Text * 0.0151) / 2

    End If

    Timer1.Enabled = True

    TestStage = 1
    MonitorStage = 1


    ' retract to zero start - home if not initialized
    Status.Caption = "Initializing..."
    DoEvents
    Call ApiRetract

wait_Retract:

    DoEvents

    If ApiComplete(0, "RETRACT") <> &HAABFF Then GoTo wait_Retract
```

```
    Timer1.Enabled = False

    Instruct1.Show 1

    If Main.Choice = 1 Then Exit Sub

    Timer1.Enabled = True

    ' run the product test now that everytihing is setup
    Status.Caption = "Testing Product..."
    Call ApiTest

wait_Test:

    DoEvents

    If ApiComplete(0, "TEST") <> &HAABFF Then GoTo wait_Test

    ' write test data to the open file and complete

    Status.Caption = "Storing Test Data..."

    oexdata.Description = Description.Text
    oexdata.pDate = DateText.Text
    ' only valid one right now
    oexdata.Setup = 0

    oexdata.BFcoef = BFcoef.Text
    oexdata.CBcoef = CBcoef.Text
    oexdata.TFcoef = TFcoef.Text
    oexdata.IScoef = IScoef.Text
    oexdata.SSprof = SSprof.Text

    oexdata.Hdzone = frmSurface.HeadZone.Text
    oexdata.Szone = frmSurface.ShoulderZone.Text
    oexdata.Wzone = frmSurface.WaistZone.Text
    oexdata.Hpzone = frmSurface.HipZone.Text
    oexdata.Tzone = frmSurface.ThighZone.Text
    oexdata.Fzone = frmSurface.FeetZone.Text

    For i = 1 To 10
        oexdata.Position(i) = Axis.FloatValueOf(i, "PFPOS")
        oexdata.Calibration(i) = Axis.FloatValueOf(i, "CAL")
    Next i

    For i = 12 To 20 Step 2
        oexdata.Position(i) = Axis.FloatValueOf(i, "PFPOS")
        oexdata.Calibration(i) = Axis.FloatValueOf(i, "CAL")
    Next i

    'close file after writing all project info

    Open filename For Binary Access Write As #1

    Put #1, 1, oexdata

    Close #1

    Status.Caption = "Test Complete!"
    Timer1.Enabled = False
    ProgressBar1.Value = 100
```

```
End Sub

Private Sub Command7_Click()

    Calibrate.Show 1

End Sub

Private Sub Command6_Click()

    EStop.Show 1

End Sub

Private Sub Command8_Click()

    SetDate.Show 1

End Sub

Private Sub Form_Load()

    ApiOpenPort

    DateText.Text = SetDate.Calendar1.Value

    'combobox is disabled until support can be written
    Combo1.Enabled = False

     BFcoef.Text = frmSurface.BFcoef.Text
    CBcoef.Text = frmSurface.CBcoef.Text
    TFcoef.Text = frmSurface.TFcoef.Text
    IScoef.Text = frmSurface.IScoef.Text
    SSprof.Text = frmSurface.SSprof.Text

    Option1(1).Enabled = False
    Option1(2).Enabled = False
    Option3(1).Enabled = False


End Sub

Private Sub Form_Unload(Cancel As Integer)

    ApiClosePort
    Unload SetDate

End Sub

Private Sub Option1_Click(index As Integer)


    ApiSetDrives (index)
    If Option1(2).Value = True Then
        Option1(2).Value = False
        Option1(0).Value = True
        EntryError.ErrorText.Caption = "Option not yet supported"
        EntryError.Show 1
    End If
```

```
End Sub

Private Sub Retract_Click()

Position.Show 1



End Sub

Private Sub Summary_Click()
    sum.Show 1
End Sub

Private Sub Timer1_Timer()

    If TestStage = 0 Then

    ElseIf TestStage = 1 Then

    ElseIf TestStage = 2 Then

    ElseIf TestStage = 3 Then

    End If

     If ProgressBar1.Value = 100 Then
        ProgressBar1.Value = 0
     End If
     ProgressBar1.Value = ProgressBar1.Value + 1
End Sub
```

Reset - 1

```
Private Done As Integer

Private Sub Form_Load()

    Done = False

    DoEvents

    Call ApiReset

    Done = True

End Sub

Private Sub Timer1_Timer()

    Dim Complete As Long

   If ProgressBar1.Value = 100 Then

        ProgressBar1.Value = 0
        Timer1.Enabled = False

        If Done = True Then
            Status.Caption = "Complete"
            Unload Me
            Exit Sub

        End If

        Timer1.Enabled = True

    End If

    ProgressBar1.Value = ProgressBar1.Value + 1
End Sub
```

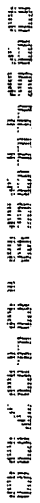
SetDate - 1

```
Option Explicit

Private Sub CancelButton_Click()

    Unload Me

End Sub

Private Sub Form_Load()

    Calendar1.Today

End Sub

Private Sub OKButton_Click()

    Main.ProjectDate = Calendar1.Value
    Main.DateText.Text = Main.ProjectDate
    Unload Me

End Sub
```

```
sum - 1

Option Explicit

Private Type Oexrecord
    ' identity stuff
    Description As String
    pDate As String
    Setup As Integer
    ' coefficients
    BFcoef As Double
    CBcoef As Double
    TFcoef As Double
    IScoef As Double
    SSprof As Double
    ' zone stuff
    Hdzone As Double
    Szone As Double
    Wzone As Double
    Hpzone As Double
    Tzone As Double
    Fzone As Double
    ' positional feedback
    Position(1 To 20) As Single
    ' calibration stuff
    Calibration(1 To 20) As Single
End Type

Private Sub mnuAbout_Click()
'User wants to see what to do in this demo.

    With CommonDialog1
        .HelpCommand = cdlHelpContext
        .HelpContext = 18
        .HelpFile = App.HelpFile
        .ShowHelp
    End With
End Sub

Private Sub mnuAboutOlectra_Click()
'User wants to see what Olectra Chart 3D is all about.

    With CommonDialog1
        .HelpCommand = cdlHelpContext
        .HelpContext = 19
        .HelpFile = App.HelpFile
        .ShowHelp
    End With
End Sub

Private Sub mnuExit_Click()
'Exit the program.

    Unload Me
End Sub



Private Sub mnuOpen_Click()
    Dim sFile As String
```

```
'     '
    With CommonDialog1
        .filename = ""
        .Flags = 0
        'To Do
        'set the flags and attributes of the
        'common dialog control
        .Filter = "Oex2000 Files (*.OEX)|*.*"
        .ShowOpen
        If Len(.filename) = 0 Then
            Exit Sub
        End If
        sFile = .filename
    End With

'    cancel = CommonDialog1.Action
    Crunchit (sFile)

End Sub
Private Sub Crunchit(sFile As String)

    Dim oexdata As Oexrecord

    Dim i As Integer

    Open sFile For Binary Access Read As #1

    Get #1, 1, oexdata

    Close #1

    Text3.Text = sFile

    Text2.Text = oexdata.Description

    BFcoef.Text = Oex2000.BFcoef

    CBcoef.Text = Oex2000.CBcoef

    TFcoef.Text = Oex2000.TFcoef

    IScoef.Text = Oex2000.IScoef

    SSprof.Text = Oex2000.SSprof


    HeadZone.Text = oexdata.Hdzone

    ShoulderZone.Text = oexdata.Szone

    WaistZone.Text = oexdata.Wzone

    HipZone.Text = oexdata.Hpzone

    ThighZone.Text = oexdata.Tzone

    FeetZone.Text = oexdata.Fzone

    frmSurface.Caption = "L&P Controls FSA Statistic Tool - " + sFile

    For i = 1 To 10
```

```
sum - 3

        'Text1(i - 1).Text = oexdata.Position(i)
    Next i
    Text1(10).Text = oexdata.Position(12)
    Text1(11).Text = oexdata.Position(14)
    Text1(12).Text = oexdata.Position(16)
    Text1(13).Text = oexdata.Position(18)
    Text1(14).Text = oexdata.Position(20)



End Sub

Private Sub Option1_Click(index As Integer)

    If index = 1 Then Frame2.Enabled = True Else: Frame2.Enabled = False


End Sub
```



```
Public Sub ApiHome()

    Axis.ChangeValue All, "HOME", "1.0"

End Sub

Public Sub ApiProduct()

    Axis.ChangeValue All, "PRODUCT", "1.0"

End Sub

Public Sub ApiTest()

    Axis.ChangeValue All, "TEST", "1.0"

End Sub

Public Sub ApiRetract()

    Axis.ChangeValue All, "RETRACT", "1.0"

End Sub

Public Sub ApiCalibrate()

    Axis.ChangeValue All, "CALIBRATE", "1.0"

End Sub

Public Sub ApiReset()

    Axis.ChangeValue All, "SWE", "0"

    Axis.PgmStop All
    Axis.Reset All
    Axis.ClearPgm (All)
    Axis.LoadPgm All, "MAIN"
    Axis.PgmRun All

    Call ApiSetDrives(0)

End Sub

Public Sub ApiEstop()

    Axis.EStop All
        DoEvents
    Axis.Reset All
        DoEvents
    Axis.ClearPgm All
        DoEvents
    Axis.LoadPgm All, "MAIN"
        DoEvents
    Axis.PgmRun All
        DoEvents


End Sub

Public Sub ApiSetDrives(Setup As Integer)
```

```
AxisControl - 1

'
' data acquisition and motion control program
'

Public Axis As New ISP

Const All = 255

Private Status As Long

' holds the trace results
Dim x(250), y(2, 250)

' number of times to retry communications
Const Retrys = 5

' windows sleep function
Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

Public Sub ApiHeadZone(Setting As String)

    Axis.ChangeValue 1, "SETTING", Setting

End Sub
Public Sub ApiShoulderZone(Setting As String)

    Axis.ChangeValue 2, "SETTING", Setting
    Axis.ChangeValue 3, "SETTING", Setting
    Axis.ChangeValue 4, "SETTING", Setting

End Sub
Public Sub ApiWaistZone(Setting As String)

    Axis.ChangeValue 5, "SETTING", Setting

End Sub
Public Sub ApiHipZone(Setting As String)

    Axis.ChangeValue 6, "SETTING", Setting
    Axis.ChangeValue 7, "SETTING", Setting
    Axis.ChangeValue 8, "SETTING", Setting
    Axis.ChangeValue 9, "SETTING", Setting

End Sub
Public Sub ApiThighZone(Setting As String)


    Axis.ChangeValue 10, "SETTING", Setting

    Axis.ChangeValue 12, "SETTING", Setting

End Sub
Public Sub ApiFeetZone(Setting As String)

    Axis.ChangeValue 14, "SETTING", Setting
    Axis.ChangeValue 16, "SETTING", Setting
    Axis.ChangeValue 18, "SETTING", Setting
    Axis.ChangeValue 20, "SETTING", Setting

End Sub
```

```
    ' which axis to deal with

    If Setup = 0 Then

        Axis.ChangeValue 1, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 2, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 3, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 4, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 5, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 6, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 7, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 8, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 9, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 10, "SWE", "1.0"
        DoEvents

        Axis.ChangeValue 12, "SWE", "1.0"

        DoEvents
        Axis.ChangeValue 14, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 16, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 18, "SWE", "1.0"
        DoEvents
        Axis.ChangeValue 20, "SWE", "1.0"

        'Axis.ChangeValue 13, "SWE", "0.0"
        'Axis.ChangeValue 15, "SWE", "0.0"
        'Axis.ChangeValue 17, "SWE", "0.0"
        'Axis.ChangeValue 19, "SWE", "0.0"

    ElseIf Setup = 1 Then

'       Axis.ChangeValue 1, "SWE", "1.0"
'       Axis.ChangeValue 2, "SWE", "1.0"
'       Axis.ChangeValue 3, "SWE", "1.0"
'       Axis.ChangeValue 4, "SWE", "1.0"
'       Axis.ChangeValue 5, "SWE", "1.0"
'       Axis.ChangeValue 6, "SWE", "1.0"
'       Axis.ChangeValue 7, "SWE", "1.0"
'       Axis.ChangeValue 8, "SWE", "1.0"
'       Axis.ChangeValue 9, "SWE", "1.0"
'       Axis.ChangeValue 10, "SWE", "1.0"
'       Axis.ChangeValue 11, "SWE", "1.0"
'       Axis.ChangeValue 12, "SWE", "1.0"

'       Axis.ChangeValue 14, "SWE", "1.0"
'       Axis.ChangeValue 16, "SWE", "1.0"
'       Axis.ChangeValue 18, "SWE", "1.0"
'       Axis.ChangeValue 20, "SWE", "1.0"
```

```
'       Axis.ChangeValue 13, "SWE", "1.0"
'       Axis.ChangeValue 15, "SWE", "1.0"
'       Axis.ChangeValue 17, "SWE", "1.0"
'       Axis.ChangeValue 19, "SWE", "1.0"


    End If


End Sub

Public Function ApiComplete(Setup As Integer, Task As String) As Long
    Status = 0
    If Setup = 0 Then

        If Axis.FloatValueOf(1, Task) = -1 Then Status = Status Or 1
        DoEvents
        If Axis.FloatValueOf(2, Task) = -1 Then Status = Status Or 2
        DoEvents
        If Axis.FloatValueOf(3, Task) = -1 Then Status = Status Or 4
        DoEvents
        If Axis.FloatValueOf(4, Task) = -1 Then Status = Status Or 8
        DoEvents
        If Axis.FloatValueOf(5, Task) = -1 Then Status = Status Or 16
        DoEvents
        If Axis.FloatValueOf(6, Task) = -1 Then Status = Status Or 32
        DoEvents
        If Axis.FloatValueOf(7, Task) = -1 Then Status = Status Or 64
        DoEvents
        If Axis.FloatValueOf(8, Task) = -1 Then Status = Status Or 128
        DoEvents
        If Axis.FloatValueOf(9, Task) = -1 Then Status = Status Or 256
        DoEvents
        If Axis.FloatValueOf(10, Task) = -1 Then Status = Status Or 512
        DoEvents

        If Axis.FloatValueOf(12, Task) = -1 Then Status = Status Or 2048
'        If Axis.FloatValueOf(13, Task) = -1 Then Home = Home Or 4096
        DoEvents
        If Axis.FloatValueOf(14, Task) = -1 Then Status = Status Or 8192
'        If Axis.FloatValueOf(15, Task) = -1 Then Home = Home Or 16384
        DoEvents
        If Axis.FloatValueOf(16, Task) = -1 Then Status = Status + 32768
'        If Axis.FloatValueOf(17, Task) = -1 Then Home = Home Or 65536
        DoEvents
        If Axis.FloatValueOf(18, Task) = -1 Then Status = Status + &H20000
'        If Axis.FloatValueOf(19, Task) = -1 Then Home = Home Or &H40000
        DoEvents
        If Axis.FloatValueOf(20, Task) = -1 Then Status = Status + &H80000

    ElseIf Setup = 1 Then

    End If

    ApiComplete = Status

End Function
Public Function ApiStartPos(Setup As Integer) As Single

    Dim fbpos(1 To 20) As Single
```

```
    Dim tr As Single
    Dim i As Integer

    If Setup = 0 Then

        For i = 1 To 20
            fbpos(i) = 0
        Next i

        For i = 1 To 10
            fbpos(i) = Axis.FloatValueOf(i, "FPOS")
            DoEvents
        Next i

        For i = 12 To 20 Step 2
            fbpos(i) = Axis.FloatValueOf(i, "FPOS")
            DoEvents
        Next i

    ElseIf Setup = 1 Then

    End If


    ' get the translation ratio
    tr = Axis.FloatValueOf(1, "TR")

    ' signal div zero error
    If tr = 0 Then


    End If

    ' translate the fbpos's to inches
    For i = 1 To 20
        fbpos(i) = fbpos(i) / tr
    Next i
```

```
End Function

Public Sub ApiOpenPort()

    Axis.OpenPort 1

    ApiReset

End Sub

Public Sub ApiClosePort()

    Axis.ClosePort
```

```
End Sub


Public Sub Capture()

    ' holds the results
    Dim x(250), y(2, 250)

    Dim Axis As Integer
    Dim jup As Integer
    Dim jlow As Integer

    Dim ct As Integer
    Dim ys As Single
    Dim ymin As Single

    ' desired axis number
    Axis = 1

    ' what to grab (jup = first set of data, jlow = second set of data)
    '
    ' 0 = analog input (ADC0)
    ' 1 = target position (TPOS)
    ' 2 = target velocity
    ' 3 = target accel
    ' 4 = feedback position
    ' 5 = feedback velocity
    ' 6 = position error
    ' 7 = current reference
    ' 8 = velocity error
    jup = 4
    jlow = 7

    ' total capture time in tenths of a second
    ' with ci_desired = 10, capture of 1 second total time
    ci_desired = 100

    ' whether (=1) or not (=0) to wait for the "WAIT FOR TRIGGER" step within
    ' a program
    trig = 1

    ' capture the data
    GoCap Axis, (ci_desired), (jup), (jlow), (trig)

    ' wait until the data is acquired by the drive
    While Not FinishedCap(Axis)
        DoEvents
    Wend

    ' first set of data (250 pts) into y(1,i)

    a = Cap(Axis, 0)
   ' a check should be made here to make sure len(a) = 250
   ' if not, the data did not make it over...

    Yscale Axis, 0, ct, ys, ymin
    If ys = 0 Then ys = 1

    For i = 1 To 250
        x(i) = 0.001 * (i - 1) * ci_desired / 2.5
```

```
        yyy = Asc(Mid(a, i))
        y(1, i) = yyy / ys + ymin
    Next i

    ' second set of data (250 pts) into y(2,i)

    a = Cap(Axis, 1)
    ' a check should be made here to make sure len(a) = 250
    ' if not, the data did not make it over...

    Yscale Axis, 1, ct, ys, ymin
    If ys = 0 Then ys = 1

    For i = 1 To 250
        yyy = Asc(Mid(a, i, 1))
        y(2, i) = yyy / ys + ymin
    Next i

    ' done

    Stop

End Sub

Public Function GoCap(id As Integer, ci As Integer, ct As Integer, ct2 As Integer, trig As Integ
er) As Boolean

    Dim i As Integer
    Dim a As String

    For i = 1 To Retrys

    If (trig = 0) Then
        Axis.SendPacket (id), Chr(11) + Chr(ci) + Chr(ct) + Chr(ct2)
    Else
        Axis.SendPacket (id), Chr(24) + Chr(ci) + Chr(ct) + Chr(ct2)
    End If

    a = Axis.GetPacket(1)

    If a <> "" Then
         GoCap = True
        Exit Function
    End If

    Next i

    GoCap = False

End Function

Function FinishedCap(id As Integer) As Boolean

    FinishedCap = False
    If Axis.Status(id) And 256 Then FinishedCap = True

End Function

Public Function Cap(id As Integer, thetype As Integer) As String

    Dim i As Integer
```

```
    For i = 1 To Retrys

    Axis.SendPacket (id), Chr(13) + Chr(thetype)
    Sleep 100

    Cap = Axis.GetPacket(250)
    If Cap <> "" Then
        Exit Function
    End If

    Next i

    Cap = ""

End Function

Public Function Yscale(id, which, ByRef captype As Integer, ByRef ys As Single, ByRef ym As Sing
le) As Boolean

    Dim i As Integer
    Dim a As String

    Yscale = False
    ys = 1
    ym = 0

    For i = 1 To Retrys

    Axis.SendPacket (id), Chr(12) + Chr(which)
    Sleep 50
    a = Axis.GetPacket(0)

    If a <> "" Then
        captype = Asc(Left(a, 1)) - 1
        a = Mid(a, 2)
        i = InStr(a, ",")
        If i <> 0 Then
            Yscale = True
            ys = Val(Left(a, i - 1))
            ym = Val(Mid(a, i + 1))
        End If
        Yscale = True
        Exit Function
    End If

    Next i

End Function
```

```
'*************************************************************************
'*
'* Copyright (c) 1998, KL GROUP INC.  All Rights Reserved.
'* http://www.klg.com
'*
'* This file is provided for demonstration and educational uses only.
'* Permission to use, copy, modify and distribute this file for
'* any purpose and without fee is hereby granted, provided that the
'* above copyright notice and this permission notice appear in all
'* copies, and that the name of KL Group not be used in advertising
'* or publicity pertaining to this material without the specific,
'* prior written permission of an authorized representative of
'* KL Group.
'*
'* KL GROUP MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY
'* OF THE SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED
'* TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
'* PURPOSE, OR NON-INFRINGEMENT. KL GROUP SHALL NOT BE LIABLE FOR ANY
'* DAMAGES SUFFERED BY USERS AS A RESULT OF USING, MODIFYING OR
'* DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
'*
'*************************************************************************

'This file contains a list of colors that can be used
' in any VB program.  These colors can also be found in
' the property pages of the Olectra Chart controls.

'The HEX values can also be broken down into their
' Red Green Blue (RGB) equivalents by breaking the number
' down by pairs of digits.  Here is an example:
'    ocColorTurquoise = &HD0E040
'    Red = &HD0 (208), Green = &HE0 (224), Blue = &H40 (64)

'ocDefaultColor is the same as "(Automatic)" in the property pages
Public Const ocDefaultColor As Long = &HFFFF

Public Const ocColorAliceBlue As Long = &HFFF8F0
Public Const ocColorAntiqueWhite As Long = &HD7EBFA
Public Const ocColorAquamarine As Long = &HD4FF7F
Public Const ocColorAzure As Long = &HFFFFF0
Public Const ocColorBeige As Long = &HDCF5F5
Public Const ocColorBisque As Long = &HC4E4FF
Public Const ocColorBlack As Long = &H0
Public Const ocColorBlanchedAlmond As Long = &HCDEBFF
Public Const ocColorBlue As Long = &HFF0000
Public Const ocColorBlueViolet As Long = &HE22B8A
Public Const ocColorBrown As Long = &H2A2AA5
Public Const ocColorBurlywood As Long = &H87B8DE
Public Const ocColorCadetBlue As Long = &HA09E5F

'ocColorChartreuse As Long = &H00FF7F
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorChartreuse As Long = 65407

Public Const ocColorChocolate As Long = &H1E69D2
Public Const ocColorCoral As Long = &H507FFF
Public Const ocColorCornflowerBlue As Long = &HED9564
Public Const ocColorCornsilk As Long = &HDCF8FF
Public Const ocColorCyan As Long = &HFFFF00
Public Const ocColorDarkGoldenrod As Long = &HB86B8
```

```
'ocColorDarkGreen As Long = &H006400
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorDarkGreen As Long = 25600

Public Const ocColorDarkKhaki As Long = &H6BB7BD
Public Const ocColorDarkOliveGreen As Long = &H2F6B55
Public Const ocColorDarkOrange As Long = &H8CFF
Public Const ocColorDarkOrchid As Long = &HCC3299
Public Const ocColorDarkSalmon As Long = &H7A96E9
Public Const ocColorDarkSeaGreen As Long = &H8FBC8F
Public Const ocColorDarkSlateBlue As Long = &H8B3D48
Public Const ocColorDarkSlateGray As Long = &H4F4F2F
Public Const ocColorDarkTurquoise As Long = &HD1CE00
Public Const ocColorDarkViolet As Long = &HD30094
Public Const ocColorDeepPink As Long = &H9314FF
Public Const ocColorDeepSkyBlue As Long = &HFFBF00
Public Const ocColorDodgerBlue As Long = &HFF901E
Public Const ocColorFirebrick As Long = &H2222B2
Public Const ocColorFloralWhite As Long = &HF0FAFF
Public Const ocColorForestGreen As Long = &H228B22
Public Const ocColorGainsboro As Long = &HDCDCDC
Public Const ocColorGhostWhite As Long = &HFFF8F8

'ocColorGold As Long = &H00D7FF
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorGold As Long = 55295

Public Const ocColorGoldenrod As Long = &H20A5DA
Public Const ocColorGray As Long = &HBEBEBE
Public Const ocColorGray0 As Long = &H0
Public Const ocColorGray1 As Long = &H30303
Public Const ocColorGray2 As Long = &H50505
Public Const ocColorGray3 As Long = &H80808
Public Const ocColorGray4 As Long = &HA0A0A
Public Const ocColorGray5 As Long = &HD0D0D
Public Const ocColorGray6 As Long = &HF0F0F
Public Const ocColorGray7 As Long = &H121212
Public Const ocColorGray8 As Long = &H141414
Public Const ocColorGray9 As Long = &H171717
Public Const ocColorGray10 As Long = &H1A1A1A
Public Const ocColorGray11 As Long = &H1C1C1C
Public Const ocColorGray12 As Long = &H1F1F1F
Public Const ocColorGray13 As Long = &H212121
Public Const ocColorGray14 As Long = &H242424
Public Const ocColorGray15 As Long = &H262626
Public Const ocColorGray16 As Long = &H292929
Public Const ocColorGray17 As Long = &H2B2B2B
Public Const ocColorGray18 As Long = &H2E2E2E
Public Const ocColorGray19 As Long = &H303030
Public Const ocColorGray20 As Long = &H333333
Public Const ocColorGray21 As Long = &H363636
Public Const ocColorGray22 As Long = &H383838
Public Const ocColorGray23 As Long = &H3B3B3B
Public Const ocColorGray24 As Long = &H3D3D3D
Public Const ocColorGray25 As Long = &H404040
Public Const ocColorGray26 As Long = &H424242
Public Const ocColorGray27 As Long = &H454545
Public Const ocColorGray28 As Long = &H474747
Public Const ocColorGray29 As Long = &H4A4A4A
Public Const ocColorGray30 As Long = &H4D4D4D
Public Const ocColorGray31 As Long = &H4F4F4F
```

```
Public Const ocColorGray32 As Long = &H525252
Public Const ocColorGray33 As Long = &H545454
Public Const ocColorGray34 As Long = &H575757
Public Const ocColorGray35 As Long = &H595959
Public Const ocColorGray36 As Long = &H5C5C5C
Public Const ocColorGray37 As Long = &H5E5E5E
Public Const ocColorGray38 As Long = &H616161
Public Const ocColorGray39 As Long = &H636363
Public Const ocColorGray40 As Long = &H666666
Public Const ocColorGray41 As Long = &H696969
Public Const ocColorGray42 As Long = &H6B6B6B
Public Const ocColorGray43 As Long = &H6E6E6E
Public Const ocColorGray44 As Long = &H707070
Public Const ocColorGray45 As Long = &H737373
Public Const ocColorGray46 As Long = &H757575
Public Const ocColorGray47 As Long = &H787878
Public Const ocColorGray48 As Long = &H7A7A7A
Public Const ocColorGray49 As Long = &H7D7D7D
Public Const ocColorGray50 As Long = &H7F7F7F
Public Const ocColorGray51 As Long = &H828282
Public Const ocColorGray52 As Long = &H858585
Public Const ocColorGray53 As Long = &H878787
Public Const ocColorGray54 As Long = &H8A8A8A
Public Const ocColorGray55 As Long = &H8C8C8C
Public Const ocColorGray56 As Long = &H8F8F8F
Public Const ocColorGray57 As Long = &H919191
Public Const ocColorGray58 As Long = &H949494
Public Const ocColorGray59 As Long = &H969696
Public Const ocColorGray60 As Long = &H999999
Public Const ocColorGray61 As Long = &H9C9C9C
Public Const ocColorGray62 As Long = &H9E9E9E
Public Const ocColorGray63 As Long = &HA1A1A1
Public Const ocColorGray64 As Long = &HA3A3A3
Public Const ocColorGray65 As Long = &HA6A6A6
Public Const ocColorGray66 As Long = &HA8A8A8
Public Const ocColorGray67 As Long = &HABABAB
Public Const ocColorGray68 As Long = &HADADAD
Public Const ocColorGray69 As Long = &HB0B0B0
Public Const ocColorGray70 As Long = &HB3B3B3
Public Const ocColorGray71 As Long = &HB5B5B5
Public Const ocColorGray72 As Long = &HB8B8B8
Public Const ocColorGray73 As Long = &HBABABA
Public Const ocColorGray74 As Long = &HBDBDBD
Public Const ocColorGray75 As Long = &HBFBFBF
Public Const ocColorGray76 As Long = &HC2C2C2
Public Const ocColorGray77 As Long = &HC4C4C4
Public Const ocColorGray78 As Long = &HC7C7C7
Public Const ocColorGray79 As Long = &HC9C9C9
Public Const ocColorGray80 As Long = &HCCCCCC
Public Const ocColorGray81 As Long = &HCFCFCF
Public Const ocColorGray82 As Long = &HD1D1D1
Public Const ocColorGray83 As Long = &HD4D4D4
Public Const ocColorGray84 As Long = &HD6D6D6
Public Const ocColorGray85 As Long = &HD9D9D9
Public Const ocColorGray86 As Long = &HDBDBDB
Public Const ocColorGray87 As Long = &HDEDEDE
Public Const ocColorGray88 As Long = &HE0E0E0
Public Const ocColorGray89 As Long = &HE3E3E3
Public Const ocColorGray90 As Long = &HE5E5E5
Public Const ocColorGray91 As Long = &HE8E8E8
Public Const ocColorGray92 As Long = &HEBEBEB
```

```
Public Const ocColorGray93 As Long = &HEDEDED
Public Const ocColorGray94 As Long = &HF0F0F0
Public Const ocColorGray95 As Long = &HF2F2F2
Public Const ocColorGray96 As Long = &HF5F5F5
Public Const ocColorGray97 As Long = &HF7F7F7
Public Const ocColorGray98 As Long = &HFAFAFA
Public Const ocColorGray99 As Long = &HFCFCFC

'ocColorGreen As Long = &H00FF00
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorGreen As Long = 65280

Public Const ocColorGreenYellow As Long = &H2FFFAD
Public Const ocColorHoneydew As Long = &HF0FFF0
Public Const ocColorHotPink As Long = &HB469FF
Public Const ocColorIndianRed As Long = &H5C5CCD
Public Const ocColorIvory As Long = &HF0FFFF
Public Const ocColorKhaki As Long = &H8CE6F0
Public Const ocColorLavender As Long = &HFAE6E6
Public Const ocColorLavenderBlush As Long = &HF5F0FF

'ocColorLawnGreen As Long = &H00FC7C
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorLawnGreen As Long = 64636

Public Const ocColorLemonChiffon As Long = &HCDFAFF
Public Const ocColorLightBlue As Long = &HE6D8AD
Public Const ocColorLightCoral As Long = &H8080F0
Public Const ocColorLightCyan As Long = &HFFFFE0
Public Const ocColorLightGoldenrod As Long = &H82DDEE
Public Const ocColorLightGoldenrodYellow As Long = &HD2FAFA
Public Const ocColorLightGray As Long = &HD3D3D3
Public Const ocColorLightPink As Long = &HC1B6FF
Public Const ocColorLightSalmon As Long = &H7AA0FF
Public Const ocColorLightSeaGreen As Long = &HAAB220
Public Const ocColorLightSkyBlue As Long = &HFACE87
Public Const ocColorLightSlateBlue As Long = &HFF7084
Public Const ocColorLightSlateGray As Long = &H998877
Public Const ocColorLightSteelBlue As Long = &HDEC4B0
Public Const ocColorLightYellow As Long = &HE0FFFF
Public Const ocColorLimeGreen As Long = &H32CD32
Public Const ocColorLinen As Long = &HE6F0FA
Public Const ocColorMagenta As Long = &HFF00FF
Public Const ocColorMaroon As Long = &H6030B0
Public Const ocColorMediumAquamarine As Long = &HAACD66
Public Const ocColorMediumBlue As Long = &HCD0000
Public Const ocColorMediumOrchid As Long = &HD355BA
Public Const ocColorMediumPurple As Long = &HDB7093
Public Const ocColorMediumSeaGreen As Long = &H71B33C
Public Const ocColorMediumSlateBlue As Long = &HEE687B
Public Const ocColorMediumSpringGreen As Long = &H9AFA00
Public Const ocColorMediumTurquoise As Long = &HCCD148
Public Const ocColorMediumVioletRed As Long = &H8515C7
Public Const ocColorMidnightBlue As Long = &H701919
Public Const ocColorMintCream As Long = &HFAFFF5
Public Const ocColorMistyRose As Long = &HE1E4FF
Public Const ocColorMoccasin As Long = &HB5E4FF
Public Const ocColorNavajoWhite As Long = &HADDEFF
Public Const ocColorNavyBlue As Long = &H800000
Public Const ocColorOldLace As Long = &HE6F5FD
Public Const ocColorOliveDrab As Long = &H238E6B
```

```
'ocColorOrange As Long = &H00A5FF
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorOrange As Long = 42495

'ocColorOrangeRed As Long = &H0045FF
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorOrangeRed As Long = 17919

Public Const ocColorOrchid As Long = &HD670DA
Public Const ocColorPaleGoldenrod As Long = &HAAE8EE
Public Const ocColorPaleGreen As Long = &H98FB98
Public Const ocColorPaleTurquoise As Long = &HEEEEAF
Public Const ocColorPaleVioletRed As Long = &H9370DB
Public Const ocColorPapayaWhip As Long = &HD5EFFF
Public Const ocColorPeachPuff As Long = &HB9DAFF
Public Const ocColorPeru As Long = &H3F85CD
Public Const ocColorPink As Long = &HCBC0FF
Public Const ocColorPlum As Long = &HDDA0DD
Public Const ocColorPowderBlue As Long = &HE6E0B0
Public Const ocColorPurple As Long = &HF020A0
Public Const ocColorRed As Long = &HFF
Public Const ocColorRosyBrown As Long = &H8F8FBC
Public Const ocColorRoyalBlue As Long = &HE16941
Public Const ocColorSaddleBrown As Long = &H13458B
Public Const ocColorSalmon As Long = &H7280FA
Public Const ocColorSandyBrown As Long = &H60A4F4
Public Const ocColorSeaGreen As Long = &H578B2E
Public Const ocColorSeashell As Long = &HEEF5FF
Public Const ocColorSienna As Long = &H2D52A0
Public Const ocColorSkyBlue As Long = &HEBCE87
Public Const ocColorSlateBlue As Long = &HCD5A6A
Public Const ocColorSlateGray As Long = &H908070
Public Const ocColorSnow As Long = &HFAFAFF
Public Const ocColorSpringGreen As Long = &H7FFF00
Public Const ocColorSteelBlue As Long = &HB48246
Public Const ocColorTan As Long = &H8CB4D2
Public Const ocColorThistle As Long = &HD8BFD8
Public Const ocColorTomato As Long = &H4763FF
Public Const ocColorTurquoise As Long = &HD0E040
Public Const ocColorViolet As Long = &HEE82EE
Public Const ocColorVioletRed As Long = &H9020D0
Public Const ocColorWheat As Long = &HB3DEF5
Public Const ocColorWhite As Long = &HFFFFFF

'ocColorYellow As Long = &H00FFFF
'The above would be true, but Visual Basic removes the leading zeros
Public Const ocColorYellow As Long = 65535

Public Const ocColorYellowGreen As Long = &H32CD9A
```

```
Option Explicit
Dim fsa_cb  As Integer
Dim fsa_is As Integer
Dim fsa_bf As Integer
Dim fsa_tf As Integer
Dim fsa_ss As Integer

Dim PSW As Double
Dim PHW As Double


Dim Barray(0 To 2000) As Long
Dim NewData As Variant

'Storage for easier reference later
Dim NumRows As Long
Dim NumColumns As Long
Dim Sindex As Long

Dim X1 As Double
Dim X2 As Double
Dim x3 As Double
Dim x4 As Double
Dim x5 As Double
Dim x6 As Double
Dim x7 As Double
Dim x8 As Double
Dim x9 As Double

Dim SpineData As Double

Dim Stopit As Integer
Dim Head As Integer
Dim Feet As Integer
' class dim
Public FSASum As Double
Public FSAAverage As Double
Public FSASensors As Long
Public TorsoAverage As Double
Public TorsoSensors As Integer
Public ShoulderAverage As Double
Dim ShoulderWidth As Double
Public HipAverage As Double
Public WaistAverage As Double
Dim HipMaxWidth As Double
Dim WAverageWidth As Double
Public FSAWeight As Double
Dim FSAHeight As Double
Dim FSAIspring As Double
Dim TorsoLength As Double
Dim datacall As Long


Public Function Put_FSAData(ByVal element As Long, ByVal index As Long) As Double

    Dim i As Integer
    Dim x As Double
    On Error Resume Next
    Put_FSAData = -1

    If index < 0 Then Exit Function
```

```
    If index > 2000 Then [redacted] Function

    NewData = True

    x = 0

    Barray(index) = element
    If element Then datacall = datacall + 1

    For i = 0 To index
        x = x + Barray(i)
    Next i

    Put_FSAData = x

End Function


Public Property Get CBcoef() As Variant
    On Error Resume Next
    If NewData Then
        Call Crunchit
        NewData = False
    End If
    CBcoef = fsa_cb
End Property


Public Property Get IScoef() As Variant
    On Error Resume Next

    If NewData Then
        Call Crunchit
        NewData = False
    End If
    IScoef = fsa_is
End Property


Public Property Get BFcoef() As Variant
    On Error Resume Next

    If NewData Then
        Call Crunchit
        NewData = False
    End If
    BFcoef = fsa_bf
End Property


Public Property Get TFcoef() As Variant
    On Error Resume Next

    If NewData Then
        Call Crunchit
        NewData = False
    End If
    TFcoef = fsa_tf
End Property
```

```
Public Property Get SSprof([redacted] Variant
    On Error Resume Next

    If NewData Then
        Call Crunchit
        NewData = False
    End If
    SSprof = fsa_ss
End Property

Private Sub Crunchit()

    Dim r As Integer
    Dim c As Integer
    Dim RAv As Double
    Dim avrav As Double
    Dim InARow As Integer
    Dim TorsoBottom As Integer
    Dim TorsoTop As Integer
    Dim TorsoLeft As Integer
    Dim TorsoRight As Integer
    Dim Brow As Integer
    Dim HipSum As Double
    Dim WaistSum As Double
    Dim HipSensors As Integer
    Dim WFirst As Integer
    Dim WLast As Integer
    Dim LastInARow As Integer
    Dim ShoulderSum As Double
    Dim ShoulderSensors As Integer
    Dim SLast As Integer
    Dim SFirst As Integer
    Dim Stretch As Integer
    Dim WaistSensors As Integer
    Dim DTemp As Double
    Dim TorsoCenter As Double
    Dim fsa As Integer
    Dim Center As Integer
    Dim Cfirst As Integer
    Dim darray(32, 32) As Double
    Dim delta As Double
    Dim Zeros As Integer
    Dim MCoef As Double
    Dim FCoef As Double
    Dim J As Double
    Dim U1(1 To 9) As Double
    Dim U2(1 To 9) As Double
    Dim Z1(1 To 9) As Double
    Dim Z1_2(1 To 9) As Double
    Dim Z2(1 To 9) As Double
    Dim Z2_2(1 To 9) As Double
    Dim Z1_Z2(1 To 9) As Double
    Dim Y1(1 To 9) As Double

    Dim SumU2 As Double
    Dim SumZ1_2 As Double
    Dim SumZ2_2 As Double
    Dim SumZ1_Z2 As Double
    Dim SumY1_Z1 As Double
    Dim SumY1_Z2 As Double
    Dim divisor As Double
```

```
    Dim S As Double
    Dim Lumbar As Double

    Dim CData(1 To 32, 1 To 32)
    Dim UnitMultiplier As Double

    On Error Resume Next

    UnitMultiplier = 0.392156862745098

    fsa = 0
    FSASum = 0
    FSASensors = 0

    NumColumns = 32
    NumRows = 32

    For c = 1 To NumColumns
        For r = 1 To NumRows

            DTemp = Barray(fsa) * UnitMultiplier
            CData(c, r) = DTemp
            If DTemp < 50 Then
                If DTemp Then
                    FSASum = FSASum + DTemp
                    FSASensors = FSASensors + 1
                End If
            End If
            fsa = fsa + 1
        Next r

    Next c

    If FSASensors = 0 Then GoTo error_out

    'standard stat stuff
    FSAAverage = FSASum / FSASensors

    FSAWeight = FSASum * 0.03

    ' if the person is less than 80 lbs - abort
    If FSAWeight < 80 Then GoTo error_out

    'let's have a swipe at some more stats
    ' ie height, hips, waist, & shoudlers


    'set up first cb factor..

    fsa_cb = FSAWeight * 3.5
'    If FSAWeight < 200 Then fsa_cb = 2
'    If FSAWeight < 150 Then fsa_cb = 1

    Stopit = False
    For c = 1 To NumColumns
        For r = 1 To NumRows
            If CData(c, r) Then Stopit = True
            If Stopit Then Exit For
        Next r
        If Stopit Then Exit For
    Next c
```

```
    Head = c

    Stopit = False
    For c = NumColumns To 1 Step -1
        For r = NumRows To 1 Step -1
            If CData(c, r) Then Stopit = True
            If Stopit Then Exit For
        Next r
        If Stopit Then Exit For
    Next c

    Feet = c

    FSAHeight = (2 + (Feet - Head)) * 2

    If FSAAverage = 0 Then GoTo error_out

    FSAIspring = FSAWeight / FSAAverage

    fsa_is = FSAIspring * 100

'    If FSAIspring < 8.5 Then fsa_is = 2
'    If FSAIspring < 6.5 Then fsa_is = 1

    'Find the "TorsoBottom" for use in calculations
    Stopit = False
    For c = NumColumns - 5 To 1 Step -1
        For r = 1 To NumRows
            If CData(c, r) Then
                InARow = InARow + 1
                If InARow > 12 Then Stopit = True
             Else:
                InARow = 0
            End If
            If Stopit Then
                Exit For
            End If
        Next r
        If Stopit Then Exit For
    Next c

    TorsoBottom = c + 1

    Center = 0
    Cfirst = 0
    Stopit = False
    For r = 1 To NumRows
        If CData(c, r) Then
            Center = Center + 1
            If Cfirst = 0 Then Cfirst = r
        End If
    Next r


    TorsoCenter = Cfirst + Center / 2

    'Find the "TorsoTop" for use in caluculations

    Stopit = 0
    For c = TorsoBottom - 10 To 1 Step -1
```

```
    If c > 32 Then GoTo error_out
    If c < 1 Then GoTo error_out
    For r = 1 To NumRows
        If CData(c, r) Then
            InARow = InARow + 1
            If InARow < Stopit Then Stopit = InARow
        Else: InARow = 0
        End If
    Next r
    If Stopit < 5 Then Exit For
    Stopit = 0
Next c

If c < 1 Then c = 1

TorsoTop = c

'Now that we have located TorsoBottom and TorsoTop
'Find the shoulder width by slicing the torso data lengthwise

'From the Right:


For r = 1 To NumRows / 2
    For c = TorsoTop To TorsoTop + 3
        If c > 32 Then GoTo error_out
        If c < 1 Then GoTo error_out
        If CData(c, r) Then
            InARow = InARow + 1
            If InARow > 1 Then Exit For
        Else:
            InARow = 0
        End If
    Next c
    If InARow > 1 Then Exit For
Next r
TorsoRight = r


'From the Left:

For r = NumRows To NumRows / 2 Step -1
    For c = TorsoTop To TorsoTop + 3
        If c > 32 Then GoTo error_out
        If c < 1 Then GoTo error_out
        If CData(c, r) Then
            InARow = InARow + 1
            If InARow > 1 Then Exit For
            Else: InARow = 0
            End If
    Next c
    If InARow > 1 Then Exit For
Next r

TorsoLeft = r


ShoulderWidth = ((TorsoLeft - TorsoRight) * 0.75) + 3

Brow = TorsoBottom - 4
```

```
    'Hip & Waist average

    LastInARow = 0
    InARow = 0
    WFirst = 0
    WLast = 0

    HipSum = 0
    WaistSum = 0
    HipSensors = 0
    HipMaxWidth = 0
    HipAverage = 0
    WaistAverage = 0
    WAverageWidth = 0
    WaistSensors = 0

'    Dim ct As Integer

    For c = Brow To Brow + 4
        For r = Cfirst To NumRows

            If CData(c, r) Then
                HipSum = HipSum + CData(c, r)
                HipSensors = HipSensors + 1
                InARow = InARow + 1
                If InARow > LastInARow Then
                    LastInARow = InARow
                End If
            Else: InARow = 0
            End If


            If c - 6 < 1 Then GoTo error_out
            If CData(c - 6, r) Then

                WaistSum = WaistSum + CData(c - 5, r)
                WaistSensors = WaistSensors + 1

            End If

            If c - 5 < 1 Then GoTo error_out
            If CData(c - 5, r) Then
                If WFirst = 0 Then WFirst = r
                WLast = r
            End If
        Next r
        WAverageWidth = WAverageWidth + (WLast - WFirst) * 0.75
        WFirst = 0
    Next c


    If HipSensors = 0 Then HipSensors = HipSensors + 1

    If WaistSensors = 0 Then WaistSensors = WaistSensors + 1

    HipMaxWidth = LastInARow * 0.75
    HipAverage = HipSum / HipSensors
    WaistAverage = WaistSum / WaistSensors
    WAverageWidth = WAverageWidth / 4
```

```
'Shoulder average

ShoulderSum = 0
ShoulderSensors = 0
ShoulderAverage = 0
SFirst = 32
SLast = 0
Stretch = 0
InARow = 0

For c = TorsoTop To TorsoTop + 3
    If c > 32 Then GoTo error_out
    If c < 1 Then GoTo error_out

    For r = TorsoRight To TorsoLeft
        If r > 32 Then GoTo error_out
        If r < 1 Then GoTo error_out
        If CData(c, r) Then

            ShoulderSum = ShoulderSum + CData(c, r)
            ShoulderSensors = ShoulderSensors + 1

        End If
    Next r
Next c

If ShoulderSensors = 0 Then GoTo error_out

ShoulderAverage = ShoulderSum / ShoulderSensors


TorsoAverage = 0
TorsoSensors = 0

'Calculate the average for the entire torso

For c = TorsoTop To TorsoBottom
    If c > 32 Then GoTo error_out
    If c < 1 Then GoTo error_out
    For r = TorsoRight To TorsoLeft
        If r > 32 Then GoTo error_out
        If r < 1 Then GoTo error_out
        If CData(c, r) < 80 Then
            If CData(c, r) Then
                TorsoAverage = TorsoAverage + CData(c, r)
                TorsoSensors = TorsoSensors + 1
            End If
        End If
    Next r
Next c

If TorsoSensors = 0 Then GoTo error_out

TorsoLength = (TorsoBottom - TorsoTop) * 2

TorsoAverage = TorsoAverage / TorsoSensors

'Predicted (Shoulder Width)/Weight = 0.14538 - 0.00000613*(Total mmHg) +
'0.0007852*(Average) - 0.0005343*(SWidth) - 0.0007978*(TLength)

'Predicted (Hip Width)/Weight = 0.12607 +13.358*(HWidth/Total mmHg) -
```

```
    '0.0009497*(SWidth) - [redacted]020362*(HWidth) - 0.0015309*(TL[redacted]h)

'     PSW = 0.14538 - 0.00000613 * FSASum + 0.0007852 * FSAAverage - 0.0005343 * ShoulderWidth -
0.0007978 * TorsoLength
'     PHW = 0.12607 + 13.358 * (HipMaxWidth / FSASum) - 0.0009497 * ShoulderWidth - 0.0020362 * H
ipMaxWidth - 0.0015309 * TorsoLength

'     PSW = 1 / (PSW + PHW)


    fsa_bf = (((FSAWeight / ShoulderWidth) + (FSAWeight / HipMaxWidth)) / 2) * 45


'     fsa_bf = (ShoulderAverage + HipAverage) * 10

'     If PSW < 7.5 Then fsa_bf = 3
'     If PSW < 6.5 Then fsa_bf = 2
'     If PSW < 5.5 Then fsa_bf = 1


    Lumbar = WaistAverage

    fsa_tf = Lumbar * 10

'     If Lumbar < 35 Then fsa_tf = 2
'     If Lumbar < 25 Then fsa_tf = 1

    Lumbar = WaistAverage

    fsa_ss = 0
    If Lumbar > 15 Then fsa_ss = 1
    If Lumbar > 20 Then fsa_ss = 2
    If Lumbar > 23 Then fsa_ss = 3
    If Lumbar > 28 Then fsa_ss = 4
    If Lumbar > 32 Then fsa_ss = 5
    If Lumbar > 38 Then fsa_ss = 6
    If Lumbar > 42 Then fsa_ss = 7
    If Lumbar > 45 Then fsa_ss = 8
    If Lumbar > 50 Then fsa_ss = 9


    GoTo end_sub

error_out:
    fsa_cb = 0
    fsa_is = 0
    fsa_bf = 0
    fsa_tf = 0
    fsa_ss = 5
end_sub:

    Exit Sub


 End Sub
```

```
'*************************************************************************
'*
'* Copyright (c) 1998, KL GROUP INC.  All Rights Reserved.
'* http://www.klg.com
'*
'* This file is provided for demonstration and educational uses only.
'* Permission to use, copy, modify and distribute this file for
'* any purpose and without fee is hereby granted, provided that the
'* above copyright notice and this permission notice appear in all
'* copies, and that the name of KL Group not be used in advertising
'* or publicity pertaining to this material without the specific,
'* prior written permission of an authorized representative of
'* KL Group.
'*
'* KL GROUP MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY
'* OF THE SOFTWARE, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED
'* TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR
'* PURPOSE, OR NON-INFRINGEMENT. KL GROUP SHALL NOT BE LIABLE FOR ANY
'* DAMAGES SUFFERED BY USERS AS A RESULT OF USING, MODIFYING OR
'* DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
'*
'*************************************************************************
Option Explicit

Public Type TRACKER
    Original As Boolean
    Linear As Boolean
End Type

Public Type POINTAPI
    x As Long
    y As Long
End Type

#If Win32 Then
    Public Declare Function CreatePen Lib "gdi32" (ByVal nPenStyle As Long, ByVal nWidth As Long
, ByVal crColor As Long) As Long
    Public Declare Function DeleteObject Lib "gdi32" (ByVal hObject As Long) As Long
    Public Declare Function GetDC Lib "user32" (ByVal hWnd As Long) As Long
    Public Declare Function ReleaseDC Lib "user32" (ByVal hWnd As Long, ByVal hDc As Long) As Lo
ng
    Public Declare Function Rectangle Lib "gdi32" (ByVal hDc As Long, ByVal X1 As Long, ByVal Y1
 As Long, ByVal X2 As Long, ByVal Y2 As Long) As Long
    Public Declare Function SelectObject Lib "gdi32" (ByVal hDc As Long, ByVal hObject As Long)
As Long
    Public Declare Function SetROP2 Lib "gdi32" (ByVal hDc As Long, ByVal nDrawMode As Long) As
Long
#Else
    Public Declare Function CreatePen Lib "gdi" (ByVal nPenStyle As Integer, ByVal nWidth As Int
eger, ByVal crColor As Long) As Integer
    Public Declare Function DeleteObject Lib "gdi" (ByVal hObject As Integer) As Integer
    Public Declare Function GetDC Lib "user" (ByVal hWnd As Integer) As Integer
    Public Declare Function Rectangle Lib "gdi" (ByVal hDc As Integer, ByVal X1 As Integer, ByVa
l Y1 As Integer, ByVal X2 As Integer, ByVal Y2 As Integer) As Integer
    Public Declare Function ReleaseDC Lib "user" (ByVal hWnd As Integer, ByVal hDc As Integer) A
s Integer
    Public Declare Function SelectObject Lib "gdi" (ByVal hDc As Integer, ByVal hObject As Integ
er) As Integer
    Public Declare Function SetROP2 Lib "gdi" (ByVal hDc As Integer, ByVal nDrawMode As Integer)
 As Integer
#End If
```

```
Private FSAType As String
Private FSAReserve As Long
Private CalFile1 As String
Private MapFlag As Integer
Private NumArrays As Integer
Private Width As Integer
Private Height As Integer
Public UnitMultiplier As Double
Private Units As String
Private Label As String
Private SizeOfFrame As Long
Public NumberofFrames As Integer
Public Junk As String
Public FSAData(0 To 2000) As Byte



Sub ReadFile(filename)

    Open filename For Binary Access Read As #1
    FSAType = String(6, " ")
    CalFile1 = String(48, " ")

    Get #1, 1, FSAType
    Get #1, , FSAReserve
    Get #1, , CalFile1
    Get #1, , MapFlag
    If MapFlag Then

    End If

    Get #1, , NumArrays
    Get #1, , Width
    Get #1, , Height
    Get #1, , UnitMultiplier
    Units = String(6, " ")
    Label = String(32, " ")
    Get #1, , Units
    Get #1, , Label

    If NumArrays = 2 Then

    End If
    Get #1, , SizeOfFrame
    Get #1, , NumberofFrames
    Junk = String(10, " ")
    Get #1, , Junk
    FSASum = 0
    FSAWeight = 0
    FSASensors = 0
    For i = 0 To SizeOfFrame - 9
        Get #1, , FSAData(i)
    Next i

    Close #1
End Sub

'Sub NextFrame(FileName)

'    Open FileName For Binary Access Read As #1
```

VistaFile - 2

```
'    For i = NextFrame To [redacted]Frame + SizeOfFrame - 9
'        Get #1, i, FSAData(1)
'    Next i

'    Close #1
'End Sub
```

Calibrate - 1

```
Option Explicit

Private Sub CancelButton_Click()
    'call ApiAbort
    Unload Me
End Sub

Private Sub Form_Load()

    Call ApiCalibrate

End Sub

Private Sub OKButton_Click()
    Unload Me
End Sub

Private Sub Timer1_Timer()

    Dim Complete As Variant

    Status.Caption = "Calibrating"

    If ProgressBar1.Value = 100 Then

        ProgressBar1.Value = 0
        Timer1.Enabled = False
        Call GetCals
        Complete = ApiComplete(0, "CALIBRATE")
        If Complete = &HAABFF Then
            Status.Caption = "Complete"
            Exit Sub
        End If
        Timer1.Enabled = True

    End If

    ProgressBar1.Value = ProgressBar1.Value + 1

End Sub

Private Sub GetCals()

    Dim i As Integer

    For i = 0 To 19

    ' need a good way to filter out unused axes
    ' for now - hard code it for back lying

        If i <> 12 And i <> 14 And i <> 16 And i <> 18 And i <> 10 Then

            Text1(i).Text = Axis.ValueOf(i + 1, "CAL")

        End If
    Next i

End Sub
```

EntryError - 1

```
Option Explicit

Private Sub OKButton_Click()
    Unload Me
End Sub
```

```
FileError - 1

Private Sub Accept_Click()
    Main.Choice = 0
    Unload Me
End Sub

Private Sub Cancel_Click()
    Main.Choice = 1
    Unload Me
End Sub
```

```
EStop - 1

Private Done As Integer

Private Sub Form_Load()

    Done = False

    Call ApiEstop

    Done = True

End Sub

Private Sub Timer1_Timer()

   Dim Complete As Long

   If ProgressBar1.Value = 100 Then

        ProgressBar1.Value = 0
        Timer1.Enabled = False

        If Done = True Then
            Status.Caption = "Test Complete"
            Unload Me
        End If

        Timer1.Enabled = True

   End If

   ProgressBar1.Value = ProgressBar1.Value + 1
End Sub
```